УДК 519.175

О. В. Расин

АЛГОРИТМ ПРОВЕРКИ ИЗОМОРФИЗМА ДЕРЕВЬЕВ ХУСИМИ

В данной работе строится полиномиальный алгоритм для проверки изоморфизма деревьев Хусими, т. е. связных графов, у которых каждый блок является либо ребром, либо циклом. Здесь под графом мы понимаем граф без петель и кратных ребер. Построенный нами алгоритм решает за полиномиальное время частный случай более общей задачи – проблемы изоморфизма графов, состоящей в том, чтобы для любых двух заданных графов определить, являются ли они изоморфными. Проблема изоморфизма графов, как хорошо известно, имеет большое теоретическое и практическое значение. Однако до сих пор не ясно, существует ли полиномиальный алгоритм ее решения в классе всех графов. Обнаружено лишь, что такие алгоритмы существуют для некоторых классов графов, например для деревьев [1] и для графов с ограниченными степенями вершин [2]. Построенный нами алгоритм в некотором роде является обобщением алгоритма Ахо–Хопкрофта–Ульмана проверки изоморфизма деревьев [1].

Несколько слов о структуре работы. В разделе 1 приводятся вспомогательные алгоритмы, необходимые для построения основного алгоритма. В разделе 2 строится основной алгоритм – алгоритм проверки изоморфизма деревьев Хусими. Здесь же дается оценка сложности этого алгоритма. В худшем случае он работает за время $O(n^2)$, где n – порядок дерева Хусими.

1. Предварительные сведения

Определение 1.1. *Пусть $G = (V, E)$ – граф, k – натуральное число. Произвольная функция вида $f : V \to \{1, \ldots, k\}$ называется цветной функцией графа G, а $f(v)$, где $v \in V$, называется цветом вершины v. Граф, на множестве вершин которого определена цветная функция, называется цветным графом. Для цветных графов будем использовать следующие обозначения: $G = (V, E, f)$, G_f или (G, f).*

Если граф G является циклом и на множестве его вершин определена цветная функция f, то G_f будем называть *цветным циклом.*

Определение 1.2. *Пусть $G_1 = (V_1, E_1, f_1)$, $G_2 = (V_2, E_2, f_2)$ – цветные графы. Биективное отображение $\phi : V_1 \to V_2$ называется изоморфизмом цветных графов G_1 и G_2, если выполнены условия:*

1) для любой пары вершин u и v графа G_1 их образы $\phi(u)$ и $\phi(v)$ смежны в графе G_2 тогда и только тогда, когда u и v смежны в G_1;

2) $f_1(v) = f_2(\phi(v))$ для любой вершины v графа G_1.

Практически каждый алгоритм, приводимый в этой статье, использует лексикографическую сортировку кортежей, элементами которых являются натуральных числа. Описание лексикографической сортировки и доказательство следующего утверждения можно найти в книге [1].

Предложение 1.1. *Пусть $B = (B_1, B_2, \ldots, B_n)$ – последовательность кортежей, компонентами которых являются целые числа от 0 до $k-1$, и l_i – длина кортежа B_i для каждого $i = 1, 2, \ldots, n$. Алгоритм лексикографической сортировки упорядочивает последовательность B лексикографически за время $O(l^* + k)$, где $l^* = \sum\limits_{i=1}^{n} l_i$.*

Алгоритм 1.1 (изоморфизм цветных циклов).
Вход: $G_1 = (V_1, E_1, f_1)$, $G_2 = (V_2, E_2, f_2)$ – цветные циклы.
Выход: возвращает $true$, если (G_1, f_1) и (G_2, f_2) изоморфны; возвращает $false$, если (G_1, f_1) и (G_2, f_2) не изоморфны.
1. Пусть $V_1 = \{u_1, \ldots u_m\}$, $V_2 = \{v_1, \ldots, v_n\}$,

$$E_1 = \{\{u_i, u_{i+1}\} \mid i \in \{1, \ldots, m-1\}\} \cup \{u_1, u_m\},$$
$$E_2 = \{\{v_j, v_{j+1}\} \mid j \in \{1, \ldots, n-1\}\} \cup \{v_1, v_n\}.$$

Если $|V_1| \neq |V_2|$, то G_1 и G_2 не изоморфны и алгоритм заканчивает работу, возвращая $false$. Иначе, если $|V_1| = |V_2| = n$, переходим к п. 2.
2. Пусть $B_1 = (f_2(v_1), f_2(v_2), \ldots f_2(v_n))$, $B_2 = (f_2(v_2), \ldots, f_2(v_n), f_2(v_1)), \ldots,$ $B_n = (f_2(v_n), f_2(v_1) \ldots, f_2(v_{n-1}))$ – последовательность кортежей длины n. С помощью лексикографической сортировки упорядочиваем эту последовательность лексикографически. Переходим к п. 3.
3. Рассмотрим кортеж $A = (f_1(u_1), f_1(u_2), \ldots f_1(u_n))$. Проверяем, существует ли кортеж B_i, где $i = 1, \ldots n$, совпадающий поэлементно с A. Если существует, то графы изоморфны, и алгоритм заканчивает работу, возвращая $true$. Если такого кортежа не существует, то графы не изоморфны и алгоритм заканчивает работу, возвращая $false$.

Правильность работы алгоритма очевидна.

Следующее утверждение непосредственно вытекает из предложения 1.1.

Предложение 1.2. *Пусть* $G_1 = (V_1, E_1, f_1)$, $G_2 = (V_2, E_2, f_2)$ – *цветные циклы,* $f_1 : V_1 \to \{1, \dots k\}$, $f_2 : V_2 \to \{1, \dots k\}$. *Тогда время работы алгоритма* 1.1 *равно* $O(n^2)$ *в худшем случае, где* $n = \min(|V_1|, |V_2|)$.

Рассмотрим цветные циклы $G_1 = (V_1, E_1, f_1)$ и $G_2 = (V_2, E_2, f_2)$, в каждом из которых для некоторого i существует ровно одна вершина цвета i, т. е. существует единственная вершина $v_1 \in V_1$ такая, что $f_1(v_1) = i$ и единственная вершина $v_2 \in V_2$ такая, что $f_2(v_2) = i$. В дальнейшем нам понадобится линейный алгоритм, решающий проблему изоморфизма для таких графов.

Алгоритм 1.2.
Вход: $G_1 = (V_1, E_1, f_1)$, $G_2 = (V_2, E_2, f_2)$ – цветные циклы. Существует цвет i такой, что в графе G_1 имеется единственная вершина u_1 цвета i, а в графе G_2 имеется единственная вершина v_1 цвета i.
Выход: возвращает $true$, если (G_1, f_1) и (G_2, f_2) изоморфны; возвращает $false$, если (G_1, f_1) и (G_2, f_2) не изоморфны.
1. Пусть $V_1 = \{u_1, \dots u_m\}$, $V_2 = \{v_1, \dots, v_n\}$,

$$E_1 = \{\{u_i, u_{i+1}\} \mid i \in \{1, \dots, m-1\}\} \cup \{u_1, u_m\},$$
$$E_2 = \{\{v_j, v_{j+1}\} \mid j \in \{1, \dots, n-1\}\} \cup \{v_1, v_n\}.$$

Если $|V_1| \neq |V_2|$, то G_1 и G_2 не изоморфны и алгоритм заканчивает работу, возвращая $false$. Иначе, если $|V_1| = |V_2| = n$, переходим к п. 2.
2. Пусть $B_1 = (f_2(v_1), f_2(v_2), \dots f_2(v_n))$, $B_2 = (f_2(v_1), f_2(v_n), \dots, f_2(v_2))$ и $A = (f_1(u_1), f_1(u_2), \dots f_1(u_n))$. Если A совпадает поэлементно с B_1 или B_2, то графы изоморфны и алгоритм возвращает значение $true$, в противном случае графы не изоморфны и алгоритм возвращает значение $false$.

Очевидно, время работы алгоритма равно $O(n)$, где $n = \min(|V_1|, |V_2|)$.

Пусть $P = \{G_1, G_2, \dots, G_s\}$ – набор цветных циклов и $C = \{1, \dots, k\}$ – набор цветов, в которые раскрашены вершины графов из P. Известно, что для некоторого цвета $m \in C$ в каждом графе G_j, где $j \in \{1, \dots, s\}$, существует только одна вершина цвета m.

Пусть $P_1, \dots, P_l$ – подмножества множества P такие, что цветные циклы G_{j_1} и G_{j_2} принадлежат одному и тому же P_i, где $i \in \{1, \dots, l\}$, если и только если цветные циклы G_{j_1} и G_{j_2} изоморфны. Очевидно, что $P = \cup_{i=1}^{l} P_i$ и если $i, j \in \{1, \dots, l\}$ и $i \neq j$, то $P_i \cap P_j = \emptyset$. Таким образом, P_i – класс изоморфных цветных циклов для каждого $i \in \{1, \dots, l\}$.

Следующий алгоритм строит по заданному набору цветных циклов $P = \{G_1, \dots, G_s\}$ разбиение множества P на классы изоморфных цветных циклов.

Алгоритм 1.3.
Вход: набор цветных циклов $P = \{G_1 = (V_1, E_1, f_1), \ldots, G_s = (V_s, E_s, f_s)\}$. Существует цвет m такой, что в каждом цикле G_i есть только одна вершина цвета m.
Выход: разбиение множества P на классы изоморфных цветных циклов.
1. Предполагаем, что $V_i = \{v_i^1, v_i^2, \ldots, v_i^{n_i}\}$,

$$E_i = \{\{v_i^j, v_i^{j+1}\} : 1 \le j \le n_i - 1\} \cup \{v_i^1, v_i^{n_i}\},$$

$f_i(v_i^1) = m$ для всех $i = 1, \ldots, s$. Каждому цветному циклу $G_i \in P$ ставим в соответствие два кортежа натуральных чисел: $B_i^1 = (f_i(v_i^1), f_i(v_i^2), \ldots f_i(v_i^{n_i}))$, $B_i^2 = (f_i(v_i^1), f_i(v_i^{n_i}), \ldots, f_i(v_i^2))$. Обозначим через B последовательность кортежей $(B_1^1, B_1^2, \ldots B_s^1, B_s^2)$. Переходим к п. 2.
2. Упорядочиваем B лексикографически. Упорядоченную последовательность обозначим через $\tilde{B}$. Переходим к п. 3.
3. Очевидно, что если для некоторых $1 \le i, j \le s$ совпали кортежи B_i^1 и B_j^r, то будут совпадать и кортежи B_i^2 и $B_j^{r'}$, где $r, r' \in \{1, 2\}$ и $r \ne r'$. Преобразуем последовательность $\tilde{B}$. Для каждого $1 \le i \le s$ оставляем в последовательности $\tilde{B}$ из двух кортежей B_i^1 и B_i^2 только один следующим образом. Если $B_i^1 \le B_i^2$, то из последовательности $\tilde{B}$ выбрасываем кортеж B_i^2, если же $B_i^2 < B_i^1$, то выбрасываем B_i^1. Переходим к п. 4.
4. Те циклы из P, которые представлены первым кортежем в $\tilde{B}$, включаем в множество P_1. Циклы, которые представлены вторым отличающимся кортежем, включаем в P_2 и т. д. Таким образом, получаем искомое разбиение $\{P_1, \ldots, P_l\}$ множества P.

Следующее утверждение непосредственно вытекает из предложения 1.1.

Предложение 1.3. *Пусть дан набор цветных циклов $P = \{G_1, G_2, \ldots, G_s\}$ и $C = \{1, \ldots, k\}$ – множество, элементы которого являются цветами вершин графов из P, причем существует цвет $m \in C$ такой, что в каждом цикле G_i есть только одна вершина цвета m. Алгоритм* 1.3 *строит разбиение множества P на классы изоморфных цветных циклов за время $O(\sum_{i=1}^{s} n_i + k)$, где n_i – количество вершин в графе G_i.*

2. Алгоритм для распознавания изоморфизма деревьев Хусими

Определение 2.1. *Связный граф G называется почти деревом с параметром k, если он обладает таким остовным деревом T, что каждый блок графа G содержит не более чем k не принадлежащих дереву T ребер.*

Почти деревья с параметром 1 называют также *деревьями Хусими*. Легко видеть, что каждый блок дерева Хусими является либо ребром, либо циклом.

В дальнейшем нам понадобится следующее понятие.

Определение 2.2. *Пусть G – связный граф, $\{B_1, \ldots, B_s\}$ – множество его блоков, $\{c_1, \ldots, c_l\}$ – множество его точек сочленения. Граф $bc(G)$, вершинами которого являются элементы из $\{B_1, \ldots, B_s, c_1 \ldots, c_l\}$ и две вершины которого смежны, если одна соответствует блоку B_i, а другая точке сочленения c_j, причем $c_j \in B_i$, называется графом блоков и точек сочленения графа G.*

Нетрудно показать, что граф блоков и точек сочленения любого графа G является деревом и что вершинами степени 1 могут быть только вершины, которые соответствуют блокам графа G. Граф блоков и точек сочленения будем в дальнейшем называть *деревом блоков и точек сочленения*.

Пусть дан граф G и $bc(G)$ – дерево блоков и точек сочленения графа G. Через C обозначим центр дерева $bc(G)$ (определение центра графа можно найти в [3]). Как известно, центр дерева состоит либо из одной, либо из двух смежных вершин. Посмотрим, что представляет собой центр графа $bc(G)$. Возможны три случая.

1. Центр состоит из одной вершины B_0, которая соответствует некоторому блоку графа G.

2. Центр состоит из одной вершины c_0, которая соответствует некоторой точке сочленения графа G.

3. Центр состоит из двух вершин B_0 и c_0, первая из которых соответствует некоторому блоку графа G, а вторая – точке сочленения.

Если для дерева $bc(G)$ выполняются случаи 1 или 2, то будем рассматривать его как корневое дерево, корнем которого является центр.

Если же для дерева $bc(G)$ выполняется случай 3, то будем рассматривать его как корневое дерево, корнем которого является вершина центра, соответствующая точке сочленения.

Вершины степени 1 в $bc(G)$, не принадлежащие центру, будем называть *листьями*.

Пусть $G_1 = (V_1, E_1)$, $G_2 = (V_2, E_2)$ – деревья Хусими. При построении алгоритма для проверки изоморфизма деревьев Хусими G_1 и G_2 важную роль играют деревья блоков и точек сочленения $bc(G_1)$ и $bc(G_2)$. Естественно имеет смысл рассматривать только следующие три случая.

A. Центр $bc(G_1)$ состоит из одной вершины, которая соответствует блоку графа G_1. Центр $bc(G_2)$ состоит из одной вершины, которая соответствует блоку графа G_2.

B. Центр $bc(G_1)$ состоит из одной вершины, которая соответствует некоторой точке сочленения графа G_1. Центр $bc(G_2)$ состоит из одной вершины, которая соответствует некоторой точке сочленения графа G_2.

C. Центры $bc(G_1)$ и $bc(G_2)$ состоят из двух вершин.

Если для центров деревьев $bc(G_1)$ и $bc(G_2)$ ни один из этих случаев не имеет место, то это означает, что деревья Хусими G_1 и G_2 не изоморфны.

Перейдем к изложению алгоритма для распознавания изоморфизма деревьев Хусими. В ходе своей работы алгоритм приписывает метки (натуральные числа) вершинам деревьев $bc(G_1)$ и $bc(G_2)$. Он делает это таким образом, что корням деревьев будут приписаны одни и те же метки, если и только если графы G_1 и G_2 изоморфны. Сначала приписываются метки узлам деревьев, находящимся на самом нижнем уровне, затем, используя эти метки, приписываем метки вершинам, находящимся на уровень выше и т. д., пока не дойдем до корней. Необходимо иметь в виду, что сопоставление меток узлам, которые соответствуют точкам сочленения, и узлам, которые соответствуют блокам, осуществляется по-разному.

Пусть мы приписали метки вершинам, которые находятся на расстоянии $k+1$ от корня. Если вершины, расположенные на расстоянии k от корня, соответствуют точкам сочленения, то каждой из этих вершин ставим в соответствие упорядоченный по возрастанию кортеж меток ее сыновей. Вершинам приписываем одинаковые метки тогда и только тогда, когда кортежи, сопоставленные им, имеют одинаковую длину и совпадают поэлементно.

В том случае, если вершины, расположенные на расстоянии k от корня, соответствуют блокам, поступаем по-другому. Пусть p – вершина дерева блоков и точек сочленения, находящаяся на расстоянии k от корня, а $B = \{v_1, v_2, \dots v_m\}$ – блок, которому она соответствует. Допустим, что при помечивании вершин, находящихся на расстоянии $k+1$ от корня, использовалось t меток. Имеет смысл выделить следующие три случая:

1) p является *внутренней* вершиной дерева, т. е. p не является ни листом, ни корнем;

2) p является корнем;

3) p является листом.

Расмотрим эти случаи.

1) Пусть p является *внутренней* вершиной дерева. Обозначим через $v_{i_1}, v_{i_2}, \dots v_{i_r}$ точки сочленения блока B, которые соответствуют сыновьям вершины p в дереве блоков и точек сочленения, а $l_1, l_2, \dots, l_r$ – метки этих сыновей. Предположим, что v_{i_0} – точка сочленения блока, которая соответствует отцу вершины p в дереве блоков и точек сочленения. Определим на

вершинах цикла B цветную функцию f следующим образом:

$$f(v) = \begin{cases} l_j, & \text{если } v = v_{i_j}, \\ t+1, & \text{если } v \neq v_{i_j},\ j \in \{0,1,\ldots,r\}, \\ t+2, & \text{если } v = v_{i_0}. \end{cases}$$

Полученный цветной цикл будем обозначать B_f. Заметим, что в нем есть только одна вершина цвета $t+2$. Каждой внутренней вершине ставим в соответствие признак *internal* (внутренняя) и цветной цикл B_f.

2) Пусть p является корнем, $v_{i_1}, v_{i_2}, \ldots v_{i_r}$ – точки сочленения блока B, которые соответствуют сыновьям вершины p в дереве блоков и точек сочленения, а $l_1, l_2, \ldots, l_r$ – метки этих сыновей. Определим на вершинах цикла B цветную функцию f следующим образом:

$$f(v) = \begin{cases} l_j, & \text{если } v = v_{i_j}, \\ t+1, & \text{если } v \neq v_{i_j}, j \in \{0,1,\ldots,r\}. \end{cases}$$

Полученный цветной цикл будем обозначать B_f. Корню p ставим в соответствие признак *root* и цветной цикл B_f.

3) Если вершина p является листом, то ставим ей в соответствие признак *leaf* и количество вершин в цикле B.

Таким образом, каждая вершина, находящаяся на расстоянии k от корня, получает признак (*internal*, *root* или *leaf*) и ей сопоставляется цветной цикл или натуральное число. Опишем, как осуществляется приписывание меток.

Если $k = 0$, то мы имеем случай 2 и всего два цветных цикла, которые соответствуют корням $bc(G_1)$ и $bc(G_2)$. Применяем алгоритм 1.1. Если эти цветные циклы изоморфны, то обоим корням приписываем метку 1, в противном случае приписываем им различные метки.

Пусть $k > 0$, $p_1, p_2, \ldots p_d$ – вершины, находящиеся на расстоянии k от корня, причем первые d_1 из них, $d_1 \leq d$, являются внутренними. Сначала приписываем метки вершинам $p_1, \ldots, p_{d_1}$. С помощью алгоритма 1.3 разбиваем блоки $B_1, \ldots, B_{d_1}$, которые им соответствуют на классы, изоморфных цветных циклов $P_1, \ldots, P_{\max}$. Если $B_i \in P_j$, то вершине p_i приписываем метку j. Таким образом, каждой внутренней вершине, расположенной на расстоянии k от корня, будет сопоставлено натуральное число из диапазона $1, \ldots, \max$. Упорядочиваем по возрастанию длины циклов, которые соответствуют листьям: $p_{d_1+1}, \ldots, p_d$. Вершинам, соответствующим циклам наименьшей длины, приписываем метку $\max + 1$. Вершинам, которым соответствуют циклы со следующей по порядку длиной, отличной от наименьшей, приписываем метку $\max + 2$ и т.д.

Алгоритм 2.1 (изоморфизм деревьев Хусими).
Вход: $G_1 = (V_1, E_1)$, $G_2 = (V_2, E_2)$ – деревья Хусими.

Выход: возвращает $true$, если G_1 и G_2 изоморфны; возвращает $false$, если G_1 и G_2 не изоморфны.

1. Поиском в глубину находим блоки и точки сочленения графов G_1 и G_2 (см. [3]). Строим для каждого из этих графов деревья блоков и точек сочленения $bc(G_1)$ и $bc(G_2)$.

2. Находим центры деревьев $bc(G_1)$ и $bc(G_2)$ (см. [4]). Если имеет место один из случаев A или B (см. с. 130), то объявляем центры деревьев $bc(G_1)$ и $bc(G_2)$ их корнями, превращая тем самым эти деревья в корневые. Если имеет место случай C (см. с. 131), то объявляем корнями деревьев $bc(G_1)$ и $bc(G_2)$ вершины их центров, которые отвечают точкам сочленения графов G_1 и G_2 соответственно. Если ни один из этих случаев не имеет место, то алгоритм заканчивает работу, возвращая $false$. Переходим к п. 3.

3. Находим h_1 – высоту дерева $bc(G_1)$ и h_2 – высоту дерева $bc(G_2)$. Переходим к п. 4.

4. Если $h_1 \neq h_2$, то $bc(G_1)$ и $bc(G_2)$ не изоморфны, а значит, и графы G_1 и G_2 не изоморфны и алгоритм завершает свою работу, возвращая $false$. В противном случае, если $h_1 = h_2$, то $h := h_1$.

5. Если $h = 0$, то переходим к п. 6. Иначе переходим к п. 7.

6. Графы G_1 и G_2 являются циклами. Если $|V_1| = |V_2|$, то графы изоморфны и алгоритм завершает работу, возвращая $true$, в противном случае графы не изоморфны и алгоритм завершает работу, возвращая $false$.

7. $k := h$ (k будет пробегать значения от h до 0). Переходим к п. 8.

8. Если $k = 0$ и корни деревьев $bc(G_1)$ и $bc(G_2)$ соответствуют блокам, то переходим к п. 15. В противном случае пусть $S_k^1 = \{p_1^1, \ldots, p_r^1\}$ – список всех вершин дерева $bc(G_1)$, находящихся на расстоянии k от корня. Аналогично, $S_k^2 = \{p_1^2, \ldots, p_m^2\}$ – список всех вершин дерева $bc(G_2)$, находящихся на расстоянии k от центра. Если $r \neq m$, то графы G_1 и G_2 не изоморфны и алгоритм заканчивает свою работу, возвращая $false$. Если $r = m$, то переходим к п. 9.

9. Если вершины, находящиеся на расстоянии k от корня, соответствуют блокам, то переходим к п. 10. Если вершины, находящиеся на расстоянии k от корня, соответствуют точкам сочленения, то переходим к п. 13.

10. Предположим, что вершины в списках S_k^1 и S_k^2 расположены так, что первые s вершин S_k^1 и первые s' из S_k^2 являются внутренними. Очевидно, что если $s \neq s'$, то графы не изоморфны и алгоритм заканчивает работу, возвращая $false$. Пусть $B_1^1, \ldots, B_r^1$ – блоки графа G_1, которые соответствуют вершинам из S_k^1, а $B_1^2, \ldots, B_r^2$ – блоки графа G_2, которые соответствуют вершинам из S_k^2. Согласно нашим обозначеням список

$$InternalBlocks = \{B_1^1, \ldots, B_s^1, B_1^2, \ldots, B_s^2\}$$

содержит блоки графов G_1 и G_2, которые соответствуют внутренним вершинам, а список

$$LeafBlocks = \{B^1_{s+1}, \ldots, B^1_r, B^2_{s+1}, \ldots, B^2_r\}$$

содержит блоки графов G_1 и G_2, которые соответствуют листьям. Переходим к п. 11.

11. Приписываем цвета вершинам блоков B^1_i и B^2_j, $i, j \in \{1, \ldots, s\}$ так, как это описывалось выше перед алгоритмом. С помощью алгоритма 1.3 множество блоков $P = \{B^j_i | i \in \{1, \ldots, s\},\ j \in \{1, 2\}\}$ разбиваем на классы $P_1, \ldots, P_{\max}$ изоморфных цветных циклов. Если блок B^j_i, $i \in \{1, \ldots, s\}$, $j \in \{1, 2\}$ попал в класс P_d, то вершине, соответствующей ему в дереве $bc(G_j)$, приписываем метку d. Предположим, что для помечивания вершин из $InternalBlock$ использовались метки из диапазона $1, \ldots, \max$. Переходим к п. 12.

12. Упорядочиваем циклы из $LeafBlocks$ в соответствии с их длинами по возрастанию. Вершинам, соответствующим циклам с наименьшими длинами, приписываем метку $\max + 1$. Вершинам, которым соответствует циклы со следующей длиной – $\max + 2$ и т. д. Переходим к п. 14.

13. Каждой вершине p^j_i, $i = 1, \ldots, r$, $j = 1, 2$ ставим в соответствие упорядоченный по возрастанию кортеж меток ее сыновей. С помощью лексикографической сортировки упорядочиваем набор этих кортежей. Пусть $C = \{C_1, \ldots C_r\}$ – упорядоченный набор этих кортежей. Вершинам, которые представлены первым кортежем в C, сопоставляем 1. Вершинам, которые представлены вторым отличающимся кортежем, сопоставляем 2 и т. д. Таким образом, каждая из вершин p^j_i, $i = 1, \ldots, r$, получает метку. Если $k = 0$, переходим к п. 16. Иначе переходим к п. 14.

14. $k := k - 1$. Переходим к п. 8.

15. Пусть B_1 – блок графа G_2, соответствующий корню $bc(G_1)$, а B_2 – блок графа G_2, соответствующий корню $bc(G_2)$. Приписываем цвета вершинам блоков B_1 и B_2 так, как мы описывали это выше перед алгоритмом. С помощью алгоритма 1.1 определяем, изоморфны или нет B_1 и B_2. Если да, то G_1 и G_2 изоморфны и алгоритм заканчивает работу, возвращая $true$. Если же B_1 и B_2 не изоморфны, то G_1 и G_2 не изоморфны и алгоритм заканчивает работу, возвращая $false$.

16. Если корням деревьев приписаны одинаковые метки, то G_1 и G_2 изоморфны и алгоритм заканчивает работу, возвращая $true$. Если же им приписаны разные метки, то G_1 и G_2 не изоморфны и алгоритм заканчивает работу, возвращая $false$.

Следующая теорема доказывает корректность алгоритма.

Теорема 2.1. *Деревья Хусими G_1 и G_2 изоморфны тогда и только тогда, когда алгоритм* 2.1 *возвращает true.*

Доказательство. Пусть $bc(G_1)$ и $bc(G_2)$ – деревья блоков и точек сочленения графов G_1 и G_2. Пусть $h_1(h_2)$ – высота дерева $bc(G_1)(bc(G_2))$. Если $h_1 \neq h_2$, то графы не изоморфны и алгоритм возвращает $false$.

Пусть $h_1 = h_2 = h$. Доказательство ведется индукцией по h. Если $h = 0$, то очевидно, что утверждение теоремы справедливо. Пусть теорема доказана для всех $h < k+1$, где k – некоторое натуральное число. Пусть r_1 (r_2) – корни $bc(G_1)$ (соответственно $bc(G_2)$), $s_1^1, \ldots, s_m^1$ – сыновья r_1, а $s_1^2, \ldots, s_{m'}^2$ – сыновья r_2. Если $m \neq m'$, то графы не изоморфны и алгоритм возвращает $false$ (см. п. 8). Пусть $m = m'$. Поддеревья дерева $bc(G_1)$ ($bc(G_2)$) с корнями в вершинах s_i^1 (s_j^2), $i, j = 1, \ldots, m$, являются деревьями блоков и точек сочленения порожденных подграфов графов G_1 (соответственно G_2). По предположению индукции вершины s_i^j и $s_{i'}^{j'}$, где $i, i' \in \{1, \ldots, m\}$, $j, j' \in \{1, 2\}$, получат одинаковые метки тогда и только тогда, когда подграфы, соответствующие им, изоморфны. Отсюда легко вытекает справедливость доказываемой теоремы.

Осталось оценить временную сложность алгоритма.

Теорема 2.2. *Пусть $G_1 = (V_1, E_1)$ и $G_2 = (V_2, E_2)$ – деревья Хусими, $bc(G_1)$ и $bc(G_2)$ – деревья блоков и точек сочленения графов G_1 и G_2. Если центры деревьев $bc(G_1)$ и $bc(G_2)$ соответствуют блокам G_1 и G_2, то алгоритм* 2.1 *выполнит проверку G_1 и G_2 на изоморфизм за время $T(n) = O(n^2)$ в худшем случае. Во всех остальных случаях алгоритм* 2.1 *работает за время $O(n)$.*

Доказательство. Имеет смысл рассматривать только следующие два случая: 1) каждый из центров $bc(G_1)$ и $bc(G_2)$ состоит из двух вершин или каждый из центров состоит из одной вершины, которая соответствует точке сочленения; 2) каждый из центров $bc(G_1)$ и $bc(G_2)$ состоит из одной вершины, которая соответствует блоку. Остальные случаи существенного влияния на сложность алгоритма не оказывают.

1) В качестве корня $bc(G_1)$ ($bc(G_2)$) выступает вершина центра, которая соответствует точке сочленения. Пусть $H = 2h + 1$ – высота дерева, где h – неотрицательное целое число. Введем следующие обозначения: n_i – количество вершин, находящихся на расстоянии $2i+1$ от корня; m_i – количество вершин, находящихся на расстоянии $2i$ от корня, $i = 1, \ldots, h$. Легко видеть, что $\sum\limits_{i=1}^{h} n_i$ равно числу блоков графа G_1, а $\sum\limits_{i=1}^{h} m_i$ равно количеству точек сочленения в графе G_1. Очевидно, что основное влияние на временную сложность

работы алгоритма оказывает цикл, расположенный в пп. 8–14. Время выполнения двух соседних итераций цикла не превосходит $O(m_{j+1}+\sum_{i=1}^{n_j} l_i^j+2\cdot n_j)$, где l_i^j – количество вершин в циклах, которые соответствуют вершинам, находящимся на расстоянии j от корня. Следовательно, суммарное время работы алгоритма равно $\sum\limits_{i=1}^{h} O(m_{j+1} + \sum\limits_{i=1}^{n_j} l_i^j + 2 \cdot n_j) = O(n)$.

2) В качестве корня $bc(G_1)$ ($bc(G_2)$) выступает вершина центра, которая соответствует блоку. Пусть $H = 2h$ – высота дерева, где h – натуральное число; n_i – количество вершин, находящихся на расстоянии $2i$ от корня; m_i – количество вершин, находящихся на расстоянии $2i-1$ от корня, $i = 1, \ldots, h$. Легко видеть, что $\sum\limits_{i=1}^{h} n_i - 1$ равно числу блоков графа G_1, а $\sum\limits_{i=1}^{h} m_i$ равно количеству точек сочленения графа G_1. Очевидно, как и в предыдущем случае, время работы цикла, расположенного в пп. 8–14, равно $O(n)$.

Пусть s – длина цикла, который соответствует корню дерева. Очевидно, что время работы п. 15 равно $O(s^2)$. Таким образом, если $s = O(n)$, то сложность работы алгоритма в этом случае равна $O(n^2)$.

Литература

1. Ахо Х., Хопкрофт Дж., Ульман Дж. Построение и анализ вычислительных алгоритмов. М.: Мир, 1979.

2. Hoffmann C. M. Group-Theoretics Algorithms and Graph Isomorphism. Berlin: Springer, 1982. (Lecture Notes in Computer Science. Vol. 136.)

3. Емеличев В. А., Мельников О. И., Сарванов В. И., Тышкевич Р. И. Лекции по теории графов. М.: Наука, 1990.

4. Евстигнеев В. А., Касьянов В. Н. Теория графов: алгоритмы обработки деревьев. Новосибирск: Наука, 1994.

Статья поступила 17.12.2001 г.